ulm university universität ulm

Universität Ulm | 89069 Ulm | Germany

Fakultät für Ingenieurwissenschaften, Informatik und Psychologie
Institut für Datenbanken und Informationssysteme

Entwicklung einer mobilen Anwendung zur Beantwortung studienbegleitender Fragen

Bachelorarbeit an der Universität Ulm

Vorgelegt von:
Moritz Weiger
moritz.weiger@uni-ulm.de

Gutachter:
Prof. Dr. Manfred Reichert

Betreuer:
Johannes Schobel

2016

Fassung 14. November 2016

Satz: PDF-LaTeX 2$_\varepsilon$

Kurzfassung

Studierende sind heutzutage in Vorlesungen wenig motiviert mitzuteilen, ob Lehrinhalte verstanden wurden. Dem Dozenten fehlt somit in der Vorlesungszeit ein Feedback darüber, bei welchen Lehrinhalten es einer Wiederholung bedarf und welche Lehrinhalte sofort verstanden und verinnerlicht wurden. Die Studierenden zu befragen darf jedoch nicht viel Vorlesungszeit in Anspruch nehmen, sollte sich nahtlos in die Vorlesung eingliedern und Ergebnisse sollten in Echtzeit zur Verfügung stellen. Durch ein Audience Response System können sowohl Studierende, als auch Dozenten eine Rückmeldung darüber erhalten und entsprechend handeln.

Zu diesem Zweck wurde ein bereits vorhandenens Audience Response System um eine mobile Anwendung für iOS und Android erweitert. Dafür wurde ein Anwendungsfall ausgearbeitet, um den Ablauf des genutzten Systems aufzuzeigen. Anschließend wurden bereits vorhandene Systeme untersucht und verglichen. Daraufhin wurde ein Konzept für die zu entwickelnde mobile Anwendung ausgearbeitet, in welchem Anforderungen und das zur Umsetzung verwendete Framework genauer betrachtet wurden. Dieses Konzept führte anschließend zur Umsetzung einer mobilen Anwendung als Erweiterung eines Audience Response Systems.

Danksagung

Ich danke meinem Betreuer Johannes Schobel für die Unterstützung in zahlreichen Besprechungen, in denen er für fast jedes Problem einen Lösungsweg parat hatte. Außerdem danke ich Fabian Schwab für seine Vorarbeit, welche als Basis meiner Arbeit diente.

Inhaltsverzeichnis

1

Einleitung

In Lehrveranstaltungen an Universitäten sind Studierenden oft wenig motiviert, sich zu melden oder in der Vorlesung einzubringen. Ob die entsprechenden Lehrinhalte von den Studierenden verstanden wurden, wird oftmals erst am Ende des Semesters anhand der Prüfungsleistungen ersichtlich. Dem Dozenten und auch den Studierenden fehlt während der Vorlesungszeit meist ein direktes Feedback, ob Lehrinhalte verstanden wurden, oder nicht.

Für den Dozenten gibt es hier mehrere Möglichkeiten, dieses Problem zu lösen. Er könnte zum Beispiel Fragen mündlich an alle Studierende richten. Anschließend melden sich diejenigen Studierende, welche glauben, die richtige Antwort zu wissen. Hierbei bestehen jedoch mehrere Probleme: Was ist mit den Studierenden, welche sich nicht melden? Wissen diese die Antwort nicht, sind sie zu wenig motiviert, die richtige Antwort abzugeben oder befürchten sie, etwas Falsches zu sagen und melden sich aufgrund dieser Befürchtung erst gar nicht, auch wenn ihre Antwort richtig wäre? Um Antworten auf diese Fragen zu finden, wäre eine anonyme Befragung möglichst aller Studierenden nötig. Zwar wäre auch eine schriftliche Befragung möglich, um alle Studierende zu erreichen, diese wäre jedoch mit erheblichem zeitlichem Aufwand, sowohl bei der Durchführung, als auch bei der Auswertung, verbunden.

Abhilfe schaffen hier sogenannte Audience Response Systeme (ARS). Bei Systemen dieser Art werden über elektronische Geräte anonym vom Dozenten vorbereitete Fragen von Studierenden beantwortet und in Echtzeit ausgewertet. Solche Audience Response Systeme existieren bereits in verschiedenen Varianten. Hardwarebasierte Systeme kosten jedoch oft viel Geld. Softwarebasierte Varianten sind oftmals günstiger, bieten aber nicht immer den Funktionsumfang, welche eine Universität mit gut besuchten

Vorlesungen verlangt und lassen sich aufgrund von oftmals geschlossenen Systemen auch nicht einfach erweitern.

In der Abschlussarbeit [1] wurde bereits ein solches System auf Basis einer Webplattform und einem Plugin für Microsoft PowerPoint als offenes System umgesetzt. Dieses soll nun um eine mobile Anwendung zur Beantwortung der Fragen für die meistgenutzten mobilen Betriebssysteme erweitert werden. Die Offenheit des Systems soll dabei gewahrt werden, sodass das System jederzeit um weitere Komponenten und Funktionen erweitert werden kann.

1.1 Struktur der Arbeit

Kapitel 2 stellt einen Anwendungsfall für den Einsatz des Systems vor. Dieser zeigt auf, wie ein Audience Response System in einer Universität genutzt werden könnte. Außerdem findet eine Betrachtung statt, welche Komponenten im vorhandenen System bereits umgesetzt wurden und welche Möglichkeiten es heute gibt, eine mobile Anwendung zu entwickeln. In Kapitel 3 werden bereits vorhandene Audience Response Systeme betrachtet und überprüft, inwiefern diese, die im Anwendungsfall beschriebenen Aufgaben, erfüllen können. Anschließend werden in Kapitel 4 die Anforderungen an die Eigenentwicklung genauer skizziert. Außerdem wird das verwendete Framework, welches bei der Eigenentwicklung verwendet werden soll, genauer betrachtet. In Kapitel 5 wird die Implementierung und Umsetzung vorgestellt, wobei betrachtet wird, wie die Anforderungen umgesetzt wurden. Zuletzt wird in Kapitel 6 eine Zusammenfassung gegeben und betrachtet, welche Erweiterungen des Systems in Zukunft noch denkbar wären.

2

Grundlagen

In diesem Kapitel wird auf die Grundlagen des Audience Response Systems eingegangen. Unter anderem wird ein Anwendungsfall aufgezeigt. Dieser ist bereits als Voraussetzung anzusehen, welche Funktionen das Audience Response System möglich machen und die bei dieser Arbeit entwickelte mobile Anwendung unterstützen sollen. Auch wird darauf im späteren Verlauf der Arbeit wiederholt eingegangen, um zu prüfen, welche Funktionen auch von bereits vorhandenen Projekten umgesetzt wurden. Es wird des Weiteren auf die Rahmenbedingungen der Vorarbeit eingegangen. Am Ende des Kapitels steht noch der Vergleich unterschiedlicher Methoden zur Entwicklung mobiler Anwendungen. Anhand dessen wird eine Auswahl getroffen, welche Methode für die Umsetzung der im Rahmen dieser Arbeit entwickelten mobilen Anwendung gewählt wurde.

2.1 Anwendungsfall

Der Anwendungsfall ist in Abbildung 2.1 grafisch dargestellt. Zum besseren Verständnis des Anwendungsfalls werden im Folgenden die Begriffe **Webplattform**, **Präsentationssoftware** und **Mobile Anwendung** genauer erläutert und zueinander abgegrenzt:

Webplattform: Die Webplattform bietet Dozenten oder Betreuern einer Vorlesung die Möglichkeit, Fragen und entsprechende Antworten zu erstellen. Diese Fragen werden an die Präsentationssoftware übertragen. Die Webplattform sammelt über entsprechende Schnittstellen die Antworten der Studenten ein und leitet diese wiederum an die Präsentationssoftware weiter. Sie soll mehreren Dozenten und Betreuern die Möglichkeit

bieten, Fragen und Antworten zu erstellen, einzusehen und zu bearbeiten. Hierfür ist auch ein entsprechendes Rechtemanagement nötig.

Präsentationssoftware: Dozenten beziehungsweise Betreuer nutzen eine Präsentationssoftware, um die Lehrinhalte für die Studenten entsprechend darzustellen. Die Fragen und Antworten sollen direkt in der Präsentationssoftware angezeigt werden, um einen flüssigen Ablauf der Veranstaltung zu gewährleisten. Zu den bekanntesten Präsentationssoftwares zählen Microsoft PowerPoint, Google Slides, Apple Keynote oder die Beamer-Klasse in LaTeX.

Mobile Anwendung: In der mobilen Anwendung bekommen die Studierenden die entsprechenden Fragen angezeigt und schicken auch über diese ihre Antwort ab. Die Studierenden interagieren nur durch die mobile Anwendung mit dem Gesamtsystem.

Anwendungsfall:
An einer Universität bietet ein Dozent zusammen mit einem oder mehreren Betreuern eine Vorlesung an und bereitet diese vor. Dabei möchten die Verantwortlichen eine Rückmeldung über den Lernerfolg der in der Lehrveranstaltung vermittelten Inhalte von den teilnehmenden Studierenden erhalten. So können sie zeitnah feststellen, ob die Studierenden die vermittelnden Lehrinhalte verstanden und verinnerlicht haben. Sollten Inhalte nicht verstanden worden sein, so besteht für die Verantwortlichen der Lehrveranstaltung die Möglichkeit, entsprechende Inhalte zu wiederholen oder zu vertiefen.

Dafür erstellen der Dozent beziehungsweise Betreuer in einer entsprechenden Webplattform eine Frage und die dazugehörigen Antwortmöglichkeiten (1). Diese werden anschließend in der Präsentation eingefügt (2).

In der Lehrveranstaltung nutzen die Verantwortlichen eine Präsentationssoftware, über die die Lehrinhalte den Studierenden auf einzelnen Seiten beziehungsweise Folien aufgezeigt werden. Neben den Lehrinhalten soll nun innerhalb der Präsentationssoftware eine Seite beziehungsweise Folie mit Fragen erscheinen (3). Hierbei handelt es sich um die Fragen, welche der Dozent zuvor in der Webplattform erstellt hat. Der Dozent stellt den Studierenden daraufhin eine angemessene Zeit zur Beantwortung der gestellten Frage zur Verfügung. Die Studenten sollen die Frage über eigene mobile Endgeräte, wie zum Beispiel Smartphones oder Tablets, beantworten können (4).

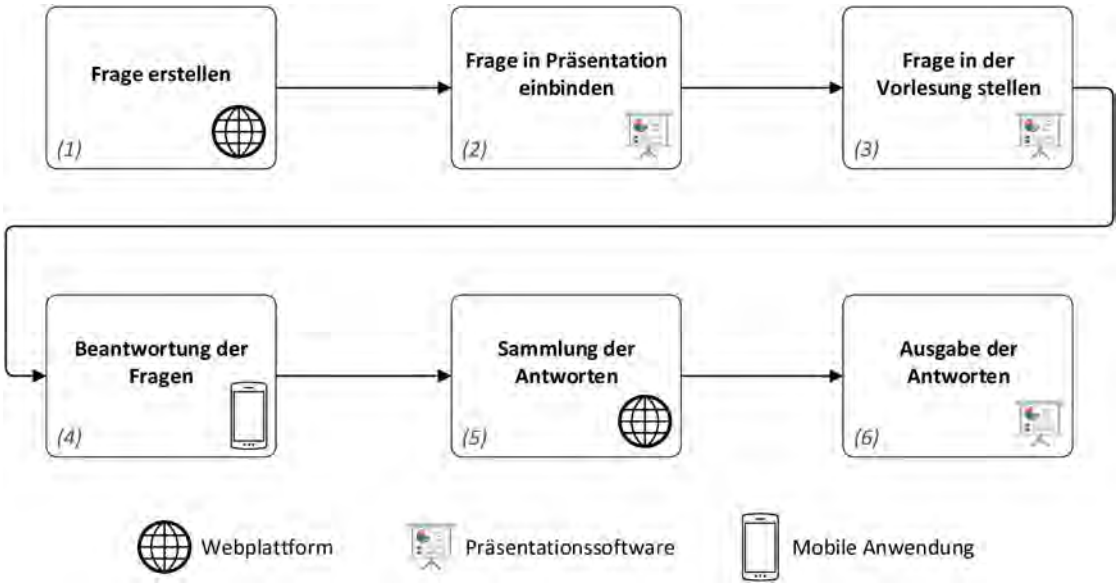


Abbildung 2.1: Allgemeiner Ablauf bei der Nutzung eines ARS

Die Anonymität soll sichergestellt werden, sodass nicht auf einzelne Studenten zurückzuführen ist, ob sie die gestellten Fragen richtig oder falsch beantwortet haben. Beim Vorgehen handelt es sich nämlich nicht um eine Leistungsabfrage, sondern lediglich um eine Feststellung des Lernerfolgs.

Die Webplattform sammelt die von den Studenten abgegebenen Antworten (5).

Nachdem der Dozent den Studenten Zeit zur Beantwortung der Fragen gegeben hat, fährt er nun in der Präsentationssoftware fort. Die Präsentationssoftware holt sich über eine Schnittstelle die Antworten der Studierenden von der Webplattform. Auf der kommenden Seite werden die Ergebnisse/Antworten der Studierenden in einem der Frage angemessenen Diagramm aufgezeigt (6). Somit ist es für die Verantwortlichen der Lehrveranstaltung und auch für die Studierenden ersichtlich, welche Lehrinhalte verstanden wurden beziehungsweise bei welchen Themen noch Defizite bestehen.

Der Dozent kann anschließend, falls die Inhalte nicht ausreichend verstanden wurden, weitere Schritte einleiten, er wiederholt entsprechende Lehrinhalte oder versucht diese durch weitere Beispiele erneut zu erklären.

2.2 Rahmenbedingungen der Vorarbeit

Als Grundlage dieser Arbeit dient die Masterthesis *Design and Implementation of an Audience Response System for Smart Mobile Devices* von Fabian Schwab von 2015 [1]. In seiner Arbeit geht er auf die grundlegenden Funktionen eines Audience Response Systems ein. Er unterscheidet zwischen hardware- und softwarebasierten Systemen und hat in seiner Arbeit bereits einen Webservice und ein Plugin für die Präsentationssoftware Microsoft PowerPoint entwickelt. Dabei unterstützt das System lediglich einen Fragetyp, nämlich die True-False-Question.

Das System wurde dabei so entwickelt, dass es sich leicht erweitern lässt. Dies wird durch einen modularen Aufbau und bekannte und unabhängige Kommunikationsprotokolle gewährleistet. Darauf wird in 4.1 genauer eingegangen.

Auch wurden bereits vorhandene ARS untersucht. Im Laufe der Zeit haben sich diese Systeme weiterentwickelt und es kamen auch neue hinzu. Deshalb werden diese in Kapitel 3 erneut untersucht und mit dem hier entwickelten System verglichen.

2.3 Entwicklung einer mobilen Anwendung

Es gibt unterschiedliche Methoden eine mobile Anwendung zu entwickeln. Diese bieten einen jeweils anderen Umfang, was die Unterstützung von Hard- und Softwarekomponenten eines mobilen Endgerätes angeht. Vor der eigentlichen Entwicklung der Anwendung müssen die erforderlichen Hard- und Softwarekomponenten festgestellt und entsprechend eine Methode zur Entwicklung gewählt werden.

Folgende Methoden sind zur Entwicklung einer mobilen Anwendung möglich:

Native Anwendung: Eine nativ entwickelte Anwendung nutzt die native Programmiersprache eines mobilen Betriebssystems. Durch die Nähe zur Systemsprache bietet die native Anwendung den größtmöglichen Funktionsumfang und kann die Ressourcen und Schnittstellen des mobilen Endgerätes optimal ausnutzen. Der Entwickler benötigt jedoch besondere Kenntnisse über gerätespezifische Entwicklungsfragen. Außerdem

benötigt der Entwickler für eine Applikation, die über mehrere Plattformen hinweg genutzt werden soll, auch die Kenntnisse über mehrere Programmiersprachen und über die technologischen Besonderheiten der einzelnen mobilen Betriebssysteme [2, 3.1].

Jedes mobile Betriebssystem nutzt zur nativen Entwicklung einer Anwendung eine andere Programmiersprache. Zur App-Entwicklung für Android wird die Programmiersprache Java genutzt. Zur Entwicklung von iOS-Apps kann seit 2014 die eigens von Apple für iOS Anwendungen entwickelte Programmiersprache Swift genutzt werden. Zuvor musste hier auf Objective-C zur Entwicklung zurückgegriffen werden. Für Microsofts mobile Betriebssysteme Windows Phone und Windows 10 Mobile wird die Programmiersprache C# zur Entwicklung von Apps genutzt.

Soll also z.B. eine Anwendung für die beiden Plattformen Android und iOS entwickelt werden, so muss eine Entwicklung sowohl in für Android mit Java, als auch in für iOS mit Swift stattfinden. Codebestandteile sind plattformübergreifend nicht kompatibel und müssen entsprechend umgeschrieben oder angepasst werden.

Webanwendung: Hierbei wird eine Anwendung lediglich in einem Browser des mobilen Endgerätes ausgeführt. Dadurch ist die Anwendung unabhängig vom mobilen Betriebssystem und kann auch auf jedem Endgerät, welches einen Browser zur Verfügung stellt, verwendet werden. Diese Anwendungen werden in HTML, CSS und Javascript erstellt. Für den Entwickler bietet dies oftmals den Vorteil, dass er bekannte Sprachen und Technologien einsetzen kann und sich nicht in eine für ihn unbekannte Sprache einarbeiten muss. Dieser Anwendungstyp bietet jedoch einen geringeren Funktionsumfang als eine nativ entwickelte Anwendung. Die Funktionen sind durch den Funktionsumfang des Browsers, in welchem das Programm ausgeführt wird, limitiert [2, 3.1].

Hybride Anwendung: Eine hybride Anwendung nutzt die Vorteile einer Webanwendung (Nutzung bekannter Sprachen und Technologien) und lässt diese nativ mit dem Betriebssystem des mobilen Endgerätes zusammenarbeiten. Eine API (application programming interface, dt: Programmierschnittstelle) ist durch das entsprechende Framework gegeben. Der Entwickler kann anschließend die Anwendungslogik durch bekannte Sprachen und Technologien wie JavaScript, HTML oder C# erstellen. Die Framework-spezifische API kümmert sich anschließend um die Kommunikation der Anwendung mit dem nati-

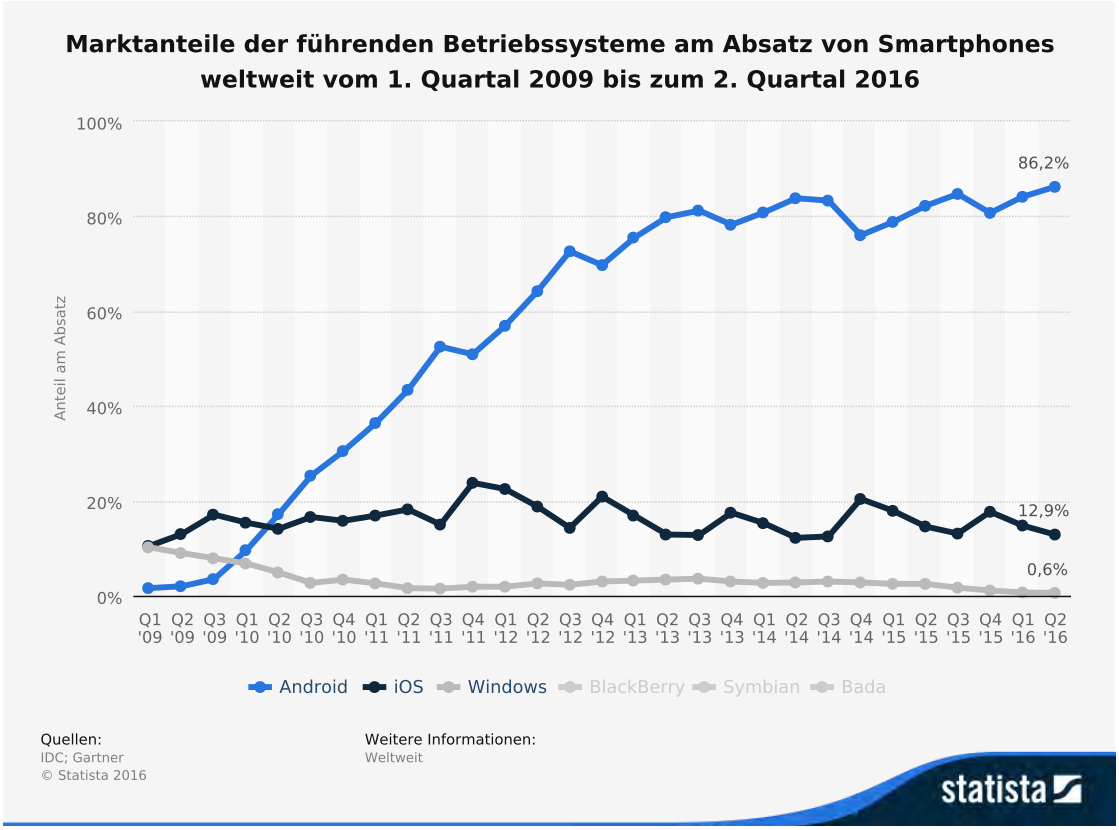


Abbildung 2.2: Marktanteile der meistgenutzten mobilen Betriebssysteme [7]

ven Betriebssystem. Somit kann ein Entwickler seinen Quellcode für alle Plattformen gemeinsam nutzen und muss nicht, wie bei einer nativen Anwendung, für jedes mobile Betriebssystem eine eigene Anwendung in einer eigenen Programmiersprache entwickeln [2, 3.1].

Mit Hilfe von diversen Frameworks bieten viele Softwarehersteller die Möglichkeiten, hybride Apps zu entwickeln. Jedes dieser Frameworks bringt unterschiedliche Vor- und Nachteile mit sich, sowie Unterstützung für verschiedene mobile Betriebssysteme. Teilweise bieten sie auch die Möglichkeit, simultan eine Anwendung für mobile Geräte und für Desktopbetriebssysteme zu entwickeln. Beispiele für solche Frameworks zur nativen Anwendungsentwicklung sind Phonegap [3] , Framework7 [4] und Ionic [5] (jeweils Enwicklung mit HTML, CSS und JavaScript) sowie Xamarin [6] (Entwicklung mit C#).

Um eine Entscheidung über die Entwicklungsmethode treffen zu können, muss auch entschieden werden, für welche (mobilen) Betriebssysteme eine Anwendung bereitgestellt werden soll. In Abbildung 2.2 sind die Marktanteile der führenden mobilen Betriebssysteme aufgeführt [7]. Android ist mit 86,2% am stärksten vertreten. Es folgt iOS mit 12,9% und Windows Phone mit 0,6% Marktanteil.

Die mobilen Betriebssysteme Bada und Symbian werden seit 2013 nicht mehr weiterentwickelt beziehungsweise unterstützt. Die Blackberry-Betriebssysteme Blackberry OS und Blackberry 10 haben lediglich einen Marktanteil von 0,1%. Deshalb wurde auf eine Darstellung dieser Betriebssysteme verzichtet und es wird im weiteren Verlauf der Arbeit nicht auf diese eingegangen.

Es sollten für die Beantwortung von studienbegleitenden Fragen möglichst viele Studierende teilnehmen können. Für die Implementierung der Applikation zur vorliegenden Arbeit wurde deshalb eine hybride Anwendung mit Hilfe des Frameworks Ionic erstellt. Der Vorteil, die Applikation und deren Quellcode für die beiden meistgenutzten mobilen Betriebssysteme Android und iOS simultan zu entwickeln, war für diese Entscheidung ausschlaggebend.

Im nächsten Kapitel werden bereits bestehende Audience Response Systeme verglichen und deren Funktionsumfang anhand des Anwendungsfalles untersucht.

3

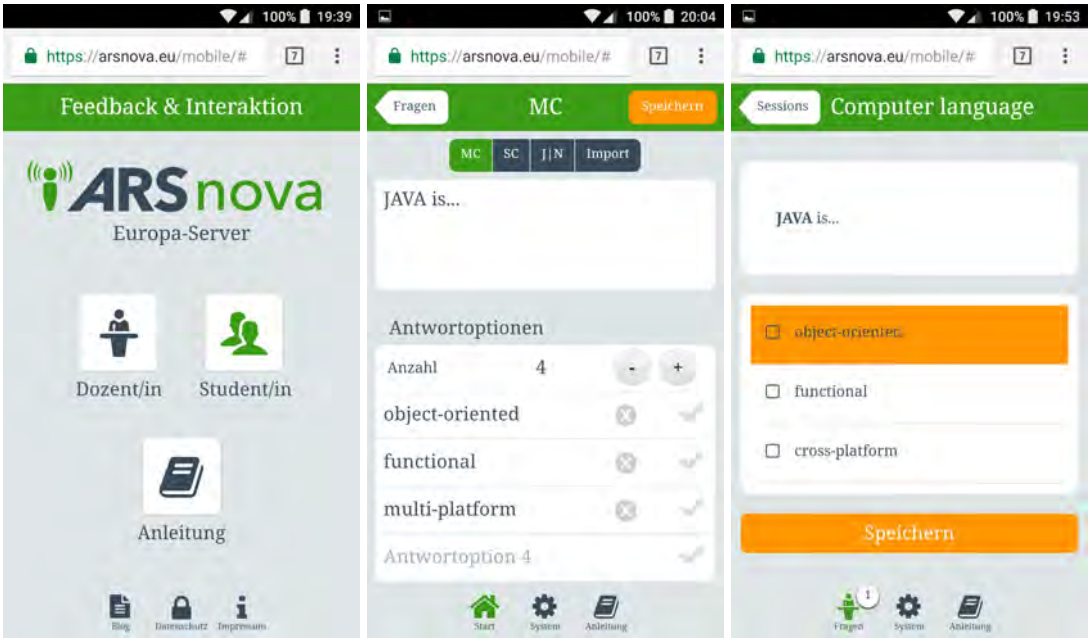
Verwandte Systeme

In diesem Kapitel stehen bereits vorhandene Audience Response Systeme im Fokus. Dabei werden deren Funktionen im Hinblick auf den in Kapitel 2.1 aufgestellten Anwendungsfall untersucht. Hier soll aufgezeigt werden, welche Anforderungen von den bereits existierenden Projekten in welchem Umfang umgesetzt werden.

3.1 ARSnova

ARSnova [8] ist ein Open-Source-Projekt der Technischen Hochschule Mittelhessen. Es ist für den Einsatz an Universitäten und Hochschulen ausgelegt und bietet eine Webplattform, über welche sowohl Dozenten Fragen erstellen, als auch Studierende diese beantworten können (siehe Abbildung 3.1 (a)). Der Dozent kann dabei mehrere Fragen erstellen (siehe Abbildung 3.1 (b)) und diese in der Reihenfolge anpassen. Dabei stehen ihm Fragetypen wie Multiple- und Single-Choice, Zwischenfragen und Live-Feedback während der Vorlesung, Evaluationsfragen und Lernkarten fürs Selbststudium zur Verfügung. Die Fragen lassn sich mit Bildern erweitern. Außerdem kann er die Antworten mit Punkten versehen, sodass Fragen und Antworten Punkte für die Beantwortung geben. Ein Login ist über die eigene Plattform, wie auch Facebook und Google+ möglich, ein Dozent kann sich jedoch auch als Gast anmelden, wobei seine Fragen dann nur temporär gespeichert werden.

Studierende können ihre Antworten ebenfalls über die Webplattform abgeben (siehe Abbildung 3.1 (c)). Dabei wählen sie zu Beginn die Rolle des Studenten und geben anschließend den bei der Erstellung einer Frage automatisch erstellten Zugangsschlüssel



(a) Rolle wählen (b) Frage erstellen (c) Frage beantworten

Abbildung 3.1: Webplattform von ARSnova (Screenshots)

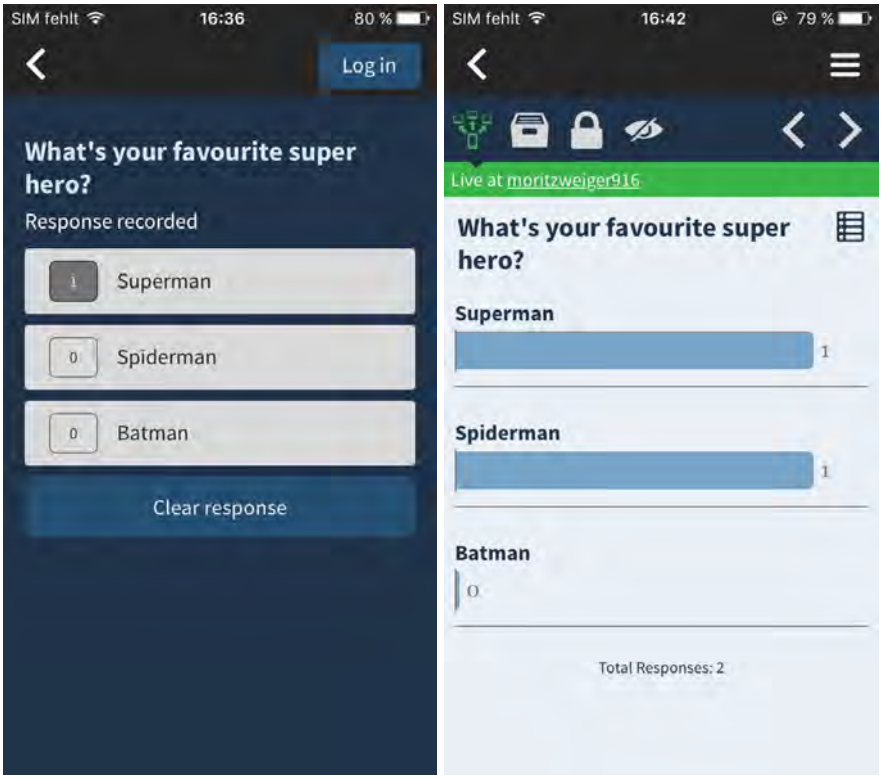
ein. Die Webplattform ist responsiv gestalten und lässt sich dadurch plattformunabhängig von jedem beliebigen Endgerät mit Browserunterstützung nutzen.

ARSnova ist Open-Source und wird stetig weiterentwickelt. Seit November 2016 gibt es den Ableger arsnova.click, welcher als spielbasierte Quiz-Anwendung mit Live-Wettbewerben und Ranglisten mit anschließender Bonusvergabe als Motivation für Studierende dienen soll, damit diese Vorlesungen genauer verfolgen oder vorbereiten.

3.2 Poll Everywhere

Poll Everywhere [9] bietet die Möglichkeit, Umfragen während einer Präsentation durchzuführen. Hauptbestandteil ist eine Webplattform, über die der Dozent Fragen erstellen kann. Es stehen ihm dabei unterschiedliche Frage- und Antworttypen zur Verfügung. Neben klassischen Antworten können auch Word Clouds aus den Antworten der Studierenden erstellt werden. Es sind auch Freitextantworten möglich, die manuell von den

Studierenden positiv oder negativ bewertet werden. Die Antworten werden dann nach den Bewertungen absteigend sortiert dargestellt. Fragen können nach dem Erstellen aktiviert oder deaktiviert werden. Nur während sie aktiviert sind, können die Teilnehmer ihre Antworten abgeben. Der Dozent hat die Möglichkeit, die anonyme Antwortabgabe für jede Frage zu aktivieren. Zur Darstellung der Frage und der abgegebenen Antworten hat der Dozent verschiedene Möglichkeiten. So können die Antworten direkt im Browser visuell dargestellt werden. Poll Everywhere bietet aber auch eine Unterstützung für gängige Präsentationssoftware. Es steht ein Plugin für Microsoft PowerPoint, Google Slides und Apple Keynote zur Verfügung, wodurch die Fragen und Antworten direkt in der entsprechenden Präsentationssoftware integriert und angezeigt werden können, ohne die Anwendung während der Präsentation wechseln zu müssen.



(a) Abgabe einer Antwort (b) Kontrolle über gestellte Frage

Abbildung 3.2: iOS-Anwendung von Poll Everywhere (Screenshots)

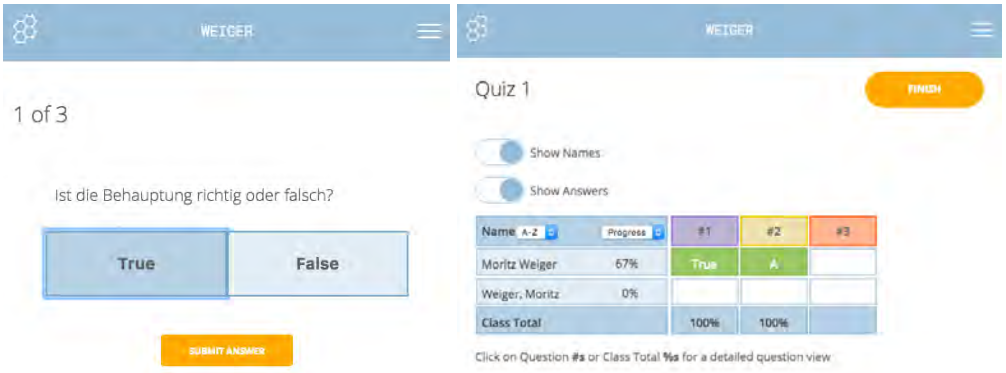
Für Studierende gibt es mehrere Möglichkeiten, ihre Antworten abzusenden. Über einen Direktlink innerhalb eines Browsers, per SMS an eine vorgegebene Mobilfunknummer, über Twitter oder über eine eigene mobile Anwendung, siehe Abbildung 3.2 (a). Über diese mobile Anwendung kann auch ein Dozent eine Frage erstellen, bereits erstellte Fragen einsehen, bearbeiten und aktivieren beziehungsweise deaktivieren, siehe Abbildung 3.2 (b). Die mobile Anwendung bietet für den Dozenten jedoch nur einen eingeschränkten Funktionsumfang im Vergleich zur Webplattform.

Poll Everywhere ist in der eingeschränkten *Higher-Ed Free* Version kostenlos, bietet jedoch auch in unterschiedlichen Preisstufen mehr Funktionsumfang. In der kostenlosen Version können lediglich 40 Antworten pro Frage abgegeben werden. Im *Instructor Plan* für 349 Dollar pro Semester bietet die Plattform bereits 400 Antworten pro Frage. Die Fragen können in beiden Versionen nicht von mehreren Präsentatoren bearbeitet werden, den sogenannten *Account Manager* bietet Poll Everywhere erst in der Preisstufe *Institution-wide* [10]. Hierfür muss zuerst ein individuelles Angebot eingeholt werden.

3.3 Socrative

Socrative [11] bietet im Vergleich zu Poll Everywhere virtuelle *Rooms*. In diesen "Räumen"werden Fragen oder auch Quizze mit mehreren Fragen angelegt. Als Fragetypen stehen True-False-Fragen, Mehrfachauswahl und Freitextantworten zur Auswahl. Für den Dozenten steht die mobile Anwendung *Socrative Teacher* für Android und iOS zur Verfügung. Diese bietet denselben Funktionsumfang wie die angebotene Webplattform, also auch das erstellen und bearbeiten von Fragen. Eine mobile Anwendung, die von Studenten und Dozenten gleichermaßen genutzt wird, wie es bei Poll Everywhere der Fall ist, gibt es nicht. Auch gibt es keine Integration in bereits vorhandene Präsentationssoftware.

Die Studenten können ihre Antworten über dieselbe Webplattform oder die mobile Anwendung *Socrative Student* abgeben. Hier wählen sie die Rolle des Studenten und geben dort den Namen des entsprechenden *Rooms* an. Anschließend wird nach dem Namen des Studierenden gefragt. Die responsive Webseite eignet sich auch zur Be-



(a) Abgabe einer Antwort (b) Einsicht der abgegebenen Antworten

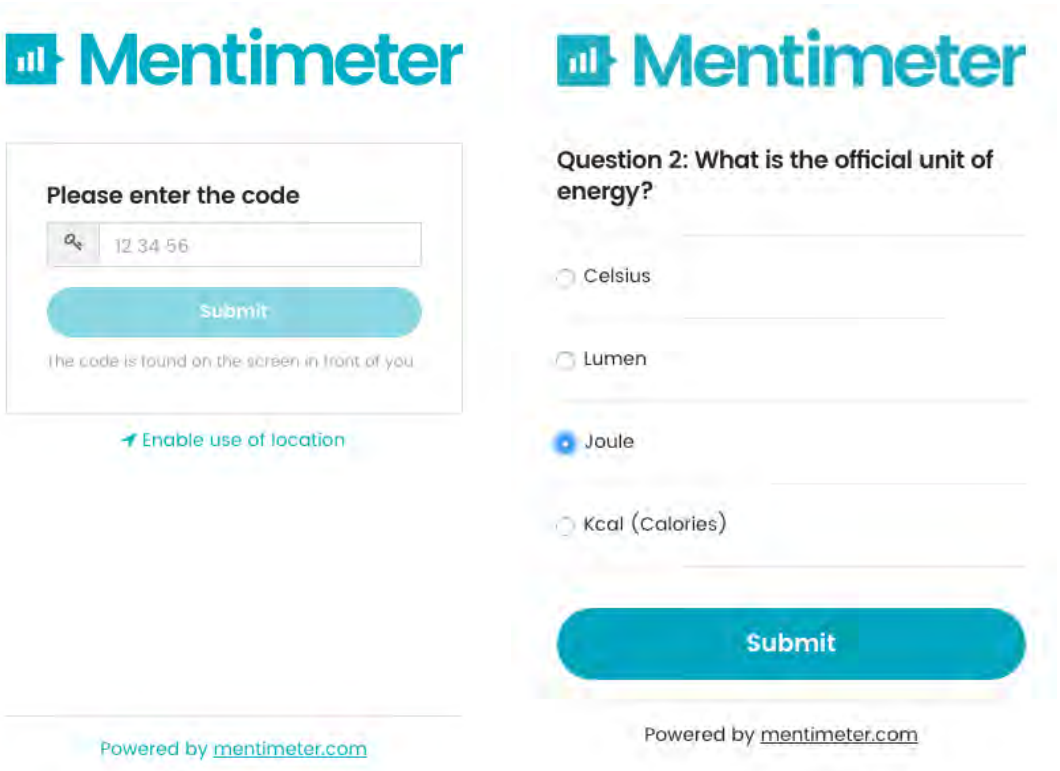
Abbildung 3.3: Browser-Ansicht von Socrative (Screenshots)

antwortung der Fragen im Browser eines mobilen Endgerätes, siehe Abbildung 3.3 (a). Dadurch ist die Installation einer mobilen Anwendung nicht zwingend notwendig.

Über die Webplattform oder die mobile Anwendung hat der Dozent die Möglichkeit, die Antworten jedes einzelnen Studierenden zu jeder Frage einzusehen (siehe Abbildung 3.3 (b)). Eine unmittelbare Ausgabe der gesammelten Antworten in einem Diagramm ist nicht möglich. Die abgegebenen Antworten können allerdings als PDF oder Excel-Datei exportiert werden. Der Name der teilnehmenden Studierenden wird auf jeden Fall übermittelt, auch wenn dieser in der Antwortübersicht ausgeblendet werden kann. Socrative bietet, wie Poll Everywhere auch, eine kostenlose und eine Pro-Variante an. In der kostenlosen Version können maximal 50 Studenten in einem öffentlichen *Room* Antworten abgeben. Für 29 US-Dollar im Jahr können bis zu 10 private oder öffentliche *Rooms* angelegt werden, in denen jeweils maximal 150 Studierende teilnehmen können [12].

3.4 Mentimeter

In Mentimeter [13] können durch den Dozenten mehrere Präsentationen mit unterschiedlichen Fragetypen angelegt werden. Hierbei stehen unter anderem Fragetypen wie Multiple Choice, Word Cloud, Likert-Skala und eine x-y-Matrix zur Verfügung. Die



(a) Teilnahme durch Eingabe des gegebenen Schlüssels

(b) Abgabe einer Antwort

Abbildung 3.4: Browser-Ansicht von Mentimeter (Screenshots)

Präsentation wird direkt in der Webplattform erstellt und für Studierende zur Beantwortung freigegeben. Die Antworten können zu jedem beliebigen Zeitpunkt während der Präsentation angezeigt werden. Eine Einbindung in vorhandene Präsentationssoftware ist allerdings hier nicht möglich, sodass zwischen Präsentations und ARS gewechselt werden muss.

Mentimeter bietet keine mobilen Anwendungen für das Beantworten der Fragen an, sondern wird über einen Link und den zur Präsentation gehörenden 6-stelligen Code aufgerufen (siehe Abbildung 3.4 (a)). Die Webseite ist responsiv gestalten und kann somit auch auf verschiedenen mobilen Endgeräten einfach genutzt werden. In der kostenlosen Version bietet Mentimeter, im Gegensatz zu Poll Everywhere und Socrative, eine unbegrenzte Anzahl an Antworten. Einschränkungen gibt es allerdings bei der

Anzahl der Fragen, die pro Präsentation auf zwei beschränkt ist. Auch der Export der Ergebnisse und die Nutzung durch mehrere Dozenten ist den käuflichen Versionen vorbehalten. Diese beginnen ab 7.99 Dollar pro Monat [14].

3.5 Abgrenzung zur Eigenentwicklung

Alle vier hier vorgestellten Lösungen bieten einen Vorteil gegenüber der Eigenentwicklung: Sie bieten die Möglichkeit, die gestellten Fragen direkt in einem Browser, also unabhängig vom genutzten Gerät und Betriebssystem, zu beantworten. So erspart man den Studenten die Installation einer eigenen Anwendung auf ihrem mobilen Endgerät und bietet zudem die Möglichkeit, die Fragen zum Beispiel auf einem Laptop zu beantworten. Möglich machen dies die responsiv entwickelten Webseiten. Die Systeme sind aufgrund der genannten Eigenschaft auch zukunftssicherer. Mobile Betriebssysteme mit geringeren Nutzerzahlen oder auch neu entwickelte Betriebssysteme können, sofern sie über einen Browser verfügen, sofort mit den Systemen interagieren.

ARSnova ist von allen vier vorgestellten Systemen am ehesten für die Nutzung an Universitäten und Hochschulen geeignet. Die Anzahl der Studierenden, welche Fragen beantworten, ist nicht limitiert. Da das System Open-Source ist, sind sämtliche unterstützte Funktionen direkt kostenlos verfügbar. Die Registrierung ist zwar möglich, aber nicht nötig.

Poll Everywhere bietet von allen aufgezeigten Lösungen den größten Funktionsumfang. Es bringt Plugins für die gängigste Präsentationssoftware mit. Außerdem ist eine Beantwortung der Fragen über unterschiedliche Kanäle, wie SMS, Twitter oder eine mobile Anwendung, möglich. ARSnova, Mentimeter und Socrative bieten keinerlei Unterstützung für bekannte Präsentationssoftware. Hier muss während einer Präsentation zwischen der genutzten Präsentationssoftware und der Webplattform gewechselt werden. Ein fließender Übergang von einer Präsentation zu einer Frage und schließlich deren Ergebnis, wie es im Anwendungsfall beschrieben ist, ist hier nicht möglich.

Socrative fällt durch eine fehlende Anonymität negativ auf. Dadurch ist das System für den in Kapitel 2.1 aufgezeigten Anwendungsfall ungeeignet. Dies zeigt jedoch auch, dass

dieses System im Grunde für ein anderes Anwendungsgebiet gedacht ist. Hierbei geht es nicht um eine anonyme Abfrage, ob die Studierenden die Inhalte verstanden haben, sondern es geht vielmehr um eine Leistungsfeststellung jedes einzelnen Studierenden. Dies kann zur Punktevergabe oder für Bonuspunkte in einer Veranstaltung genutzt werden. Es bietet sich außerdem nur für kleine Gruppen wie zum Beispiel Schulklassen an, da es auch gegen Aufpreis nur eine begrenzte Anzahl an Antworten unterstützt.

Die vier vorhandenen Systeme bieten in ihrer (kostenloser) Version keinerlei Rechtemanagement oder Benutzerverwaltung. Poll Everywhere und Mentimeter bieten dies nur gegen Aufpreis. In ARSnova und Socrative fehlt diese Funktion gänzlich. Die Eigenentwicklung bietet mit einer vorhandenen Benutzerverwaltung einen entscheidenden Vorteil: Fragen können von mehreren Dozenten erstellt, bearbeitet und genutzt werden. Dies wird durch das Rechtemanagement des vorliegenden Systems ermöglicht. Auch ist es mit diesem System möglich, eine unbegrenzte Anzahl an Studierenden an der Beantwortung der Fragen teilnehmen zu lassen.

Die Eigenentwicklung bietet jedoch den Nachteil, dass die Fragen von den Studierenden lediglich in einer eigens dafür entwickelten mobilen Anwendung, welche sie extra installieren müssen, beantwortet werden können. Es bietet jedoch offene Schnittstellen, sodass es um beliebige weitere Anwendungen und Funktionen (zum Beispiel eine responsive Webseite) weiterentwickelt werden kann.

Ergebnis Die Anforderungen können von keinem bereits vorhandenen System vollständig erfüllt werden. Deshalb wird in diesem Anwendungsfall auf eine Eigenentwicklung gesetzt.

Im nachfolgenden Kapitel wird ein Konzept für die Eigenentwicklung ausgearbeitet. Dabei werden Anforderungen an die mobile Anwendung aufgestellt und das verwendete Framework Ionic auf Funktionen, Umfang und Tools untersucht.

4

Konzeption der mobilen Anwendung

Vor der Entwicklung der mobilen Anwendung steht die Konzeption. Hierzu werden im Folgenden die Anforderungen an die Software aufgestellt, also die Aufgaben, welche die mobile Anwendung umsetzen soll. Weiter werden erste Prototypen in Form von Mockups aufgezeigt, um die Anforderungen an die zu entwickelnde Software zu verdeutlichen. Abschließend steht das Ionic Framework im Fokus. Dabei wird auf die Architektur, die mitgelieferten Tools und auf den Entwicklungsvorgang einer mobilen Anwendung mit Hilfe dieses Frameworks eingegangen.

4.1 Anforderungen

Vor der eigentlichen Entwicklung der mobilen Anwendung müssen gewisse Voraussetzungen an die Anwendung gestellt werden. Dies umfasst sowohl strukturelle Voraussetzungen, als auch den eigentlichen Funktionsumfang der Anwendung. Diese werden im Folgenden genauer erläutert.

Die strukturellen Voraussetzungen umfassen unter anderem die Kompatibilität mit der Vorarbeit, die bereits in Kapitel 2.2 beschrieben wurde. Das bisherige System wurde so aufgebaut, dass es möglichst einfach um zusätzliche Komponenten erweitert werden kann. Auch sollen bestimmte Komponenten bei Bedarf ausgetauscht werden können. Dies wird durch offene Schnittstellen möglich gemacht.

Eine weitere strukturelle Voraussetzung ist die Kompatibilität mit möglichst vielen mobilen Plattformen. Die im Zuge dieser Arbeit entwickelte mobile Anwendung sollte auf

möglichst vielen mobilen Plattformen lauffähig sein. Nur so kann eine Befragung zu einer hohen Teilnahmequote und einem ausschlaggebenden Ergebnis führen.

Zu den funktionellen Voraussetzungen gehören die Aktionen, welche von den unterschiedlichen Personengruppen ausgeführt werden. Dazu zählen die Aktionen des Dozenten beziehungsweise der Betreuer einer Vorlesung, als auch die Aktionen, welche von Studierenden innerhalb der mobilen Anwendung ausgeführt werden.

Der Dozent interagiert mit der Webplattform und der Präsentationssoftware. In der Webplattform erstellt er eine Frage mit entsprechenden Antwortmöglichkeiten. Anschließend fügt er diese in seine Präsentation ein. In der Vorarbeit (siehe Kapitel 2.2) wurden REST-basierte Designmethoden für den Austausch von Daten verwendet. Dies bietet jeder REST-fähigen Anwendung die Möglichkeit, Daten des Audience Response Systems zu empfangen oder Daten an dieses zu senden. Die Daten liegen im JSON-Format vor und werden auch so unter den verschiedenen Komponenten ausgetauscht. In der Vorarbeit wurde bereits der Fragetyp *True-False-Question* umgesetzt. Doch nicht jede Frage lässt sich auf diesen Fragetyp abbilden, weshalb der Dienst um weitere Fragetypen erweitert werden soll. Diese weiteren Fragetypen werden in Kapitel 4.2 genauer ausgeführt.

Die Studierenden interagieren lediglich über die mobile Anwendung mit dem System. Für sie gibt es sowohl grundlegende als auch optionale Anforderungen. Die grundlegenden Anforderungen müssen erfüllt werden, um einen Ablauf, wie im Anwendungsfall beschrieben, zu gewährleisten.

R1: Anmeldung für eine Vorlesung *(notwendig)* Der Studierende sucht sich eine Vorlesung aus, was auf unterschiedliche Art geschehen. Er kann die passende Vorlesung über eine Baumstruktur erreichen. Dazu geht er die entsprechenden Organizations durch, bis er bei der gesuchten Vorlesung angekommen ist. Dort meldet er sich für diese an. Ab diesem Zeitpunkt erhält er immer eine Benachrichtigung, wenn für diese Vorlesung eine neue Frage gestellt wird.

R2: Beantwortung einer Frage *(notwendig)* Sobald eine Frage für eine Vorlesung verfügbar ist, bei welchem der Studierende angemeldet ist, bekommt er eine Benachrichtigung auf sein entsprechendes mobiles Endgerät geschickt. Der Studierende soll die Fragen und Antwortmöglichkeiten auf seinem mobilen Endgerät

direkt sehen können. Anschließend beantwortet er die Frage und schickt seine Antwort ab. Es soll dabei zu keiner großen Verzögerungen, sowohl beim Anzeigen einer Fragen, als auch beim Absenden einer Antwort kommen.

R3: Abmeldung von einem Repository *(notwendig)* Der Studierende kann sich, zum Beispiel am Ende des Semesters, von einer Vorlesung abmelden. Von nun an erhält er keine weiteren Benachrichtigungen, falls zu dieser Vorlesung eine neue Frage gestellt wird.

Die optionalen Anforderungen bringen lediglich höheren Komfort für die Bedienung, aber auch eine höhere Motivation für die Studierenden, die Fragen zu beantworten.

R4: Vorlesung oder Organization suchen *(optional)* Denkbar wäre auch eine Textsuche, in welcher ein Student den Namen des gesuchten Repositories oder der gesuchten Organization eingibt. Aus den Ergebnissen wählt er dann das gesuchte Repository beziehungsweise die gesuchte Organization aus.

R5: Errungenschaften *(optional)* In der Vorarbeit wurden im Ausblick bereits Achievements (dt: Errungenschaften) als denkbare Erweiterung erwähnt. Diese stellen für die Teilnehmer einen Anreiz dar, an einer Befragung teilzunehmen, und so die Teilnahmebereitschaft deutlich zu erhöhen.

R6: Einsicht teilgenommener Fragen *(optional)* Die Studierenden können in der mobilen Anwendung alle bisher abgegebenen Antworten inklusive Fragen abrufen. Sie können somit jederzeit einsehen, bei wie vielen Fragen sie eine Antwort abgegeben haben.

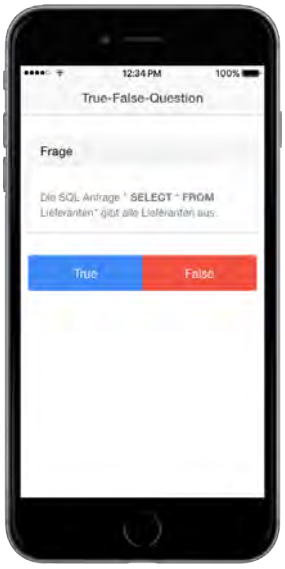
4.2 Konzeption

Die in Kapitel 4.1 aufgezeigten Anforderungen sollen in diesem Abschnitt für eine mobile Anwendung skizziert werden.

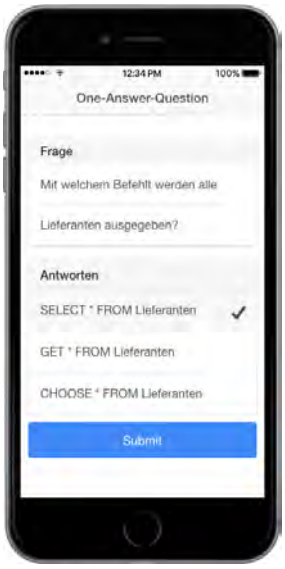
Zu den Anforderungen zählt unter anderem die Unterstützung weiterer Fragetypen. Die folgenden Fragetypen wurden dabei umgesetzt. Die dargestellten Mock-ups wurden mit Hilfe des Ionic Creators erstellt (siehe Kapitel 4.3).

True-False-Question Dieser Fragetyp bietet einem Studierenden zwei Antwortmöglichkeiten, nämlich wahr (true) oder falsch (false) (Abbildung 4.2 (a)).

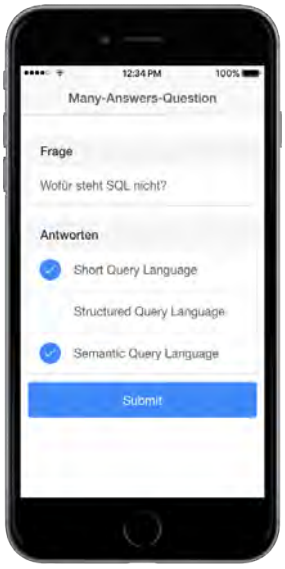
One-Answer-Question Bei diesem Fragetyp hat der Studierende die Möglichkeit, nur eine der gegebenen Antworten auszuwählen (Abbildung 4.2 (b)).



(a) Beispiel True-False-Question



(b) Beispiel One-Answer-Question



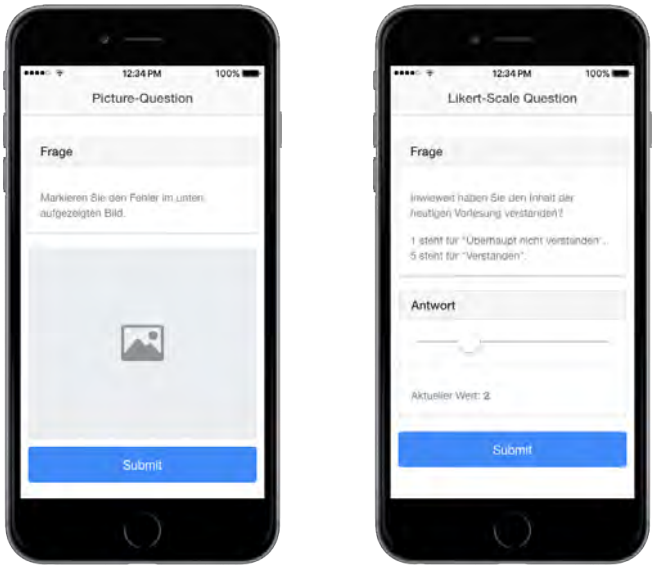
(c) Beispiel Many-Answers-Question

Abbildung 4.1: Fragetypen

Multiple-Answers-Question Dieser Fragetyp lässt den Studierenden eine oder mehrere Antwortmöglichkeiten auswählen. Im Gegensatz zu einer One-Answer Frage kann es auch mehr als eine richtige Antwort geben (Abbildung 4.2 (c)).

Picture-Question Zeigt dem Studierenden anstatt einer klassischen Textfrage ein Bild an. Anschließend wählt der Studierende einen bestimmten Punkt oder Bereich auf dem Bild aus, um zum Beispiel einen Fehler in einem Diagramm zu markieren. Die Antwort des Studierenden ist somit eine X-Y-Koordinate auf diesem Bild (Abbildung 4.2 (a)).

Liker-Scale-Question Hierbei ist dem Studierenden eine sogenannte Likert-Skala gegeben. Diese beinhaltet einen vorgegebenen Definitionsbereich, also zum Beispiel



(a) Beispiel Picture-Question

(b) Beispiel Likert-Scale-Question

Abbildung 4.2: Weitere Fragetypen

Werte von 1-5. Der Studierende kann einen dieser Werte als Antwort über einen Schieberegler wählen (Abbildung 4.2 (b)).

4.3 Architektur

Das Ionic-Framework hält eigene Tools, Kommandozeilenbefehle und weitere Hilfestellung zur Erstellung einer hybriden Anwendung bereit. Diese werden im Folgenden genauer betrachtet. Außerdem wird auf auf die Komponenten, Dateiformate und den Aufbau eines Ionic-Projektes eingegangen.

4.3.1 Das Ionic-Framwork

Die Entwicklung der mobilen Anwendung erfolgt mit dem Framework Ionic. Dieses wurde im Oktober 2013 vorgestellt [15]. Später folgten Alpha- und Betaphase und im Winter 2015 der erste Release Candidate. Ionic setzt auf das JavaScript-Framework AngularJS,

Abbildung 4.3: Das Zusammenspiel von Ionic, AngularJS, Cordova und der Nativen SDK

welches zur Entwicklung dynamischer Webanwendungen genutzt wird. Dieses baut wiederum auf Apache Cordova auf. Cordova ist für die Kommunikation zwischen den Sensordaten der mobilen Geräte und dem Framework zuständig, also die Schnittstelle zwischen hybridem Framework und nativer Unterstützung der Funktionen eines mobilen Endgerätes. Eine mobile Anwendung wird mit Ionic in HTML, CSS und JavaScript geschrieben. HTML und CSS kümmern sich dabei um Darstellung, JavaScript ist für die Programmlogik zuständig.

4.3.2 Tools und Befehle

Ionic bietet neben Kommandozeilenbefehlen, die direkt in einer Konsole eingegeben und ausgeführt werden, auch hilfreiche Tools. Diese erleichtern das Erstellen und Testen einer Ionic-Anwendung durch entsprechende GUIs. Einige dieser Tools und Kommandozeilenbefehle werden im Folgenden vorgestellt.

Ionic Creator ist ein Drag-and-Drop Tool, welches direkt im Browser ausgeführt wird. Damit werden Elemente der Anwendungsoberfläche einfach auf einen vorgegebenen Bereich gezogen, verändert, beschriftet oder auch verknüpft. Das Ergebnis kann anschließend als Ionic-Projekt exportiert oder auch im Browser direkt getestet werden. Beim Export wird bereits die komplette Ordnerstruktur eines Ionic-Projektes erstellt. In Creator kann allerdings keinerlei Anwendungslogik erstellt, sondern lediglich die Oberfläche entworfen werden. Nach dem Export kann die

Programmlogik in der eigentlichen Entwicklungslogik hinzugefügt werden. Dadurch eignen sich die Projekte sehr gut als Mock-ups, wie in Kapitel 4.2 diskutiert.

Ionic Lab ist im Gegensatz zu den bisher vorgestellten Tools keine Webplattform, sondern eine Desktop-Anwendung. Dieses steht für die aktuellen Versionen der Betriebssysteme OS X, Windows und Linux zur Verfügung und bietet es die Möglichkeit, ohne Kommandozeilenbefehle die mobile Anwendung in einem entsprechenden Emulator zu testen oder direkt auf einem angeschlossenen mobilen Endgerät auszuführen.

Ionic View ist eine mobile Anwendung für iOS und Android. Es ermöglicht den Test einer mit Ionic erstellten Anwendung direkt auf dem Endgerät. Dadurch muss nicht zuerst eine APK- beziehungsweise IPA-Datei erstellt und auf dem mobilen Endgerät installiert werden. Ionic View erlaubt den Upload unbegrenzt vieler mobiler Anwendungen.

Ionic Starters ist streng genommen kein Tool. Es handelt sich dabei um eine Sammlung an Vorlagen, in denen bestimmte Anwendungsbausteine bereits umgesetzt sind. Diese lassen sich automatisiert erstellen und anschließend vom Entwickler erweitern.

```
1 $ ionic start Anwendungsname tabs
```

Listing 4.1: Erstellung einer Ionic-Anwendung aus einer Vorlage

Mit dem Kommandozeilenbefehl aus Codebeispiel 4.1 wird die Vorlage *tabs* erstellt (siehe Abbildung 4.4. Weitere Starters sind im Ionic Market [16] zu finden. Diese bieten eine Integration von bekannten Technologien, wie zum Beispiel Google Maps, WordPress oder diverse Messenger.

Neben den Ionic Starters bietet der Market auch Plugins, die gewissen Bausteine für Anwendungen bereitstellen und sich in bestehende Projekte einfügen lassen. Auch bietet der Ionic Market vorgefertigte Themes, die das Aussehen einer erstellten Anwendung anpassen.

Ionic Serve ist ein Kommandozeilenaufruf, der die Vorschau einer mit Ionic erstellten Anwendung direkt im Webbrowser des Computers erlaubt. Dadurch kann direkt

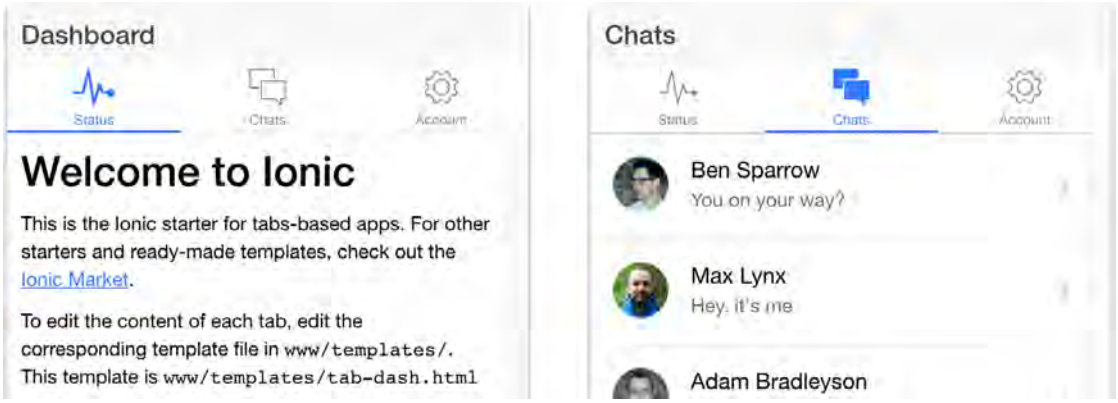


Abbildung 4.4: Nutzung der Vorlage *tabs* zur Erstellung einer Ionic-Anwendung (Screenshot der Live-Vorschau mit Ionic Serve)

verglichen werden, welche designtechnischen Unterschiede zwischen den Plattformen iOS und Android bestehen. Der Kommandozeilenaufruf ermöglicht außerdem einen Live-Reload – die Vorschau wird also automatisch aktualisiert, sobald eine Datei verändert wird. So können Änderungen unmittelbar für das jeweilige mobile Betriebssystem angesehen werden. Ein Ergebnis dieser Vorschau in einem Browser ist in Abbildung 4.5 zu sehen. Dabei bietet Ionic Serve zeitgleich eine Ansicht für iOS (1) und Android (2). Dadurch lassen sich Unterschiede der mobilen Anwendung auf beiden Plattformen direkt erkennen.

```
1 $ ionic serve --lab
```

Listing 4.2: Kommandozeilenaufruf zur Live-Vorschau im Browser

Weitere Befehle und Tools sind in der offiziellen Ionic Dokumentation [17] zu finden. Da Ionic auf AngularJS aufbaut, sind auch Befehle aus diesem Framework mit Ionic kompatibel und können somit auch in einem Ionic Projekt verwendet werden. Deshalb wird an dieser Stelle auch auf die offizielle AngularJS Dokumentation verwiesen [18].

4.3.3 Komponenten

Im Folgenden wird auf die Ordnerstruktur, Dateiformate und den allgemeinen Aufbau eines Ionic-Projekts eingegangen.

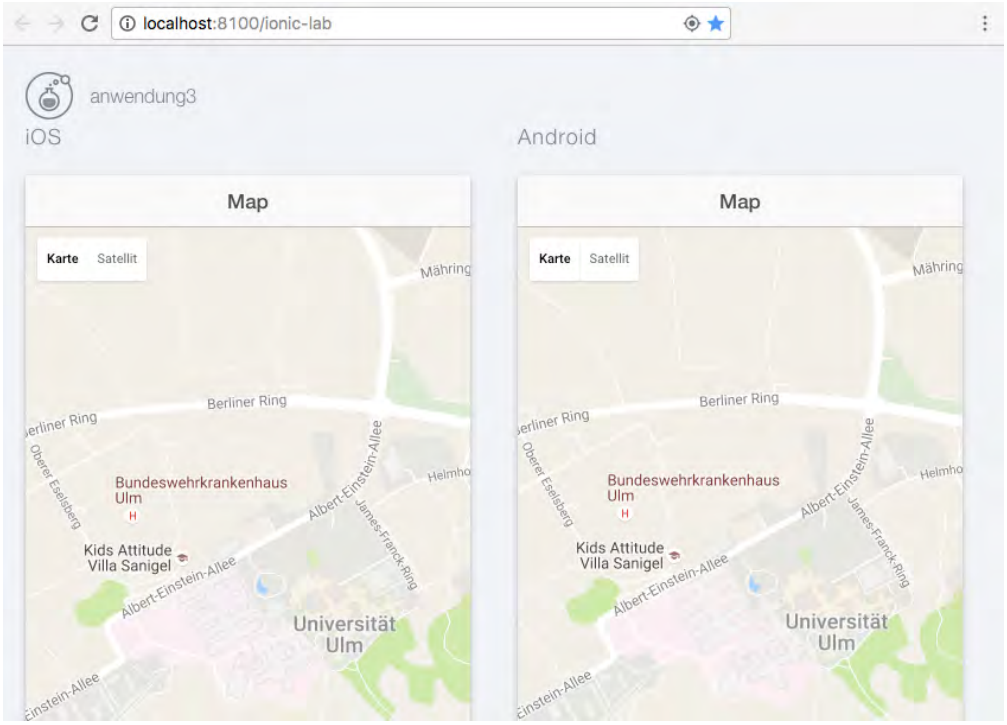


Abbildung 4.5: Nutzung des Kommandozeilenbefehls *ionic serve* zur Live-Vorschau einer Anwendung (Screenshot)

Abbildung 4.6 zeigt die Ordnerstruktur eines Ionic-Projektes. Dabei sind die für die Entwicklung relevanten Ordner `css`, `js` und `templates` hervorgehoben. Diese nutzen, wie bereits in Kapitel 2.3 erwähnt, bekannte Webtechnologien. Der Ordner `templates` (c) beinhaltet dabei die Vorlagen einzelner Views (Ansichten). Diese entsprechen hier den einzelnen Oberflächen einer mobilen Anwendung und werden über HTML spezifiziert.

Der Ordner `css` (a) enthält die in CSS gehaltenen Formatierungsvorgaben für die Oberflächen. Zusammen mit HTML ist CSS heute ein Hauptbestandteil des World Wide Web. In einem Ionic-Projekt hat CSS direkten Einfluss darauf, wie HTML-Dateien gerendert werden.

Im Ordner `js` (b) befinden sich JavaScript-Dateien. Diese haben, im Gegensatz zu HTML-Dateien, keinen direkten Einfluss auf das Aussehen einer View. Sie verändern im Falle von Ionic und AngularJS mit Hilfe von Controllern, Scopes oder Services lediglich Variablen oder Objekte einer Anwendung. JavaScript ist für die Logik innerhalb der

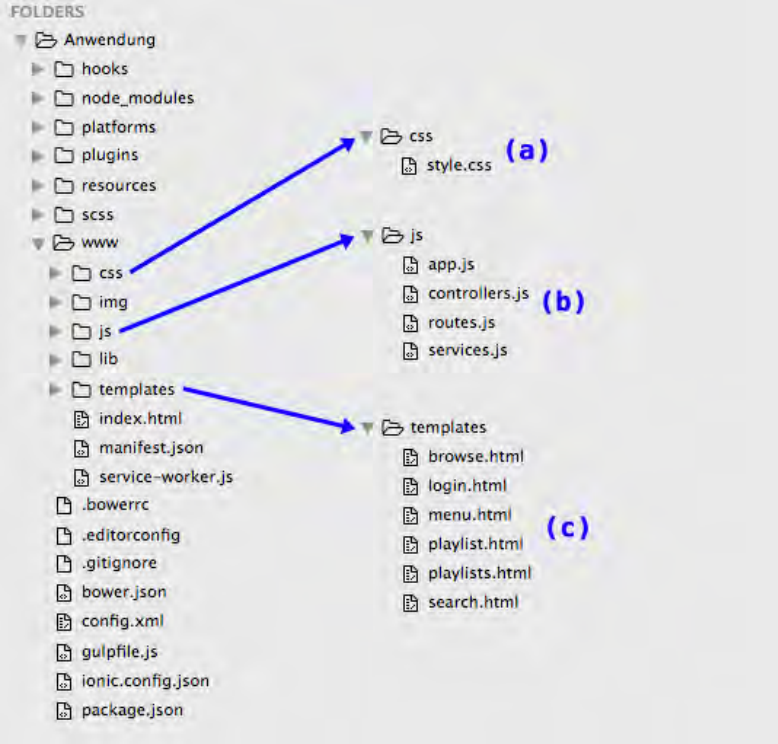


Abbildung 4.6: Die Ordnerstruktur eines Ionic-Projektes

mobilen Anwendung zuständig und wird direkt auf dem clienten und nicht auf dem Server ausgeführt. Ohne JavaScript wäre eine Ionic-Anwendung statisch und könnten nicht auf die Ein- und Ausgabe von Daten reagieren.

Die von AngularJS genutzten Methoden werden im Folgenden genauer betrachtet.

Controller *kontrollieren* die Daten einer Anwendung. Es handelt sich dabei um gewöhnliche JavaScript Objekte. Scopes werden innerhalb von Controllern initialisiert oder deren Zustand oder Wert verändert. Sollte innerhalb eines Controllers eine Funktion ausgeführt werden, so muss ihm diese übergeben werden [19].

In Listing 4.3 wird innerhalb eines Controllers einem Scope ein Text übergeben. Dies findet in einem JavaScript statt. In Listing 4.4 wird dieses Scope anschließend von einem HTML-Dokument ausgelesen und das Feld `data` damit gefüllt.

```
.controller('AppController', function($scope){
  $scope.data = "Dies ist ein Text."
})
```

Listing 4.3: Controller weist einem Scope einen Text zu

```
<ion-view>
  <ion-content>
    <!-- Controller festlegen -->
    <div data-ng-controller='AppController'>
    <!-- Inhalt eines Scope ausgeben -->
    <p> {{ data }} </p>
    </div>
  </ion-content>
<ion-view>
```

Listing 4.4: HTML-View kann über Scope auf den Text zugreifen

Scopes sind die Verbindung von Controllern und Views. Sie stellen somit die Schnittstelle zwischen statischem HTML und dynamischem JavaScript dar. Ein Scope-Objekt kann dabei ein Variable, ein Array, aber auch eine Funktion mit Ein- und Ausgabeparametern sein.

Services sind Objekte oder Funktionen. AngularJS bringt bereits eine Reihe vorgefertigter Services mit sich.

Ein Beispiel für einen durch AngularJS bereits vordefinierten Service ist der `$http`-Service, der eine entsprechende URL aufruft und das Ergebnis anschließend in einem Scope oder andere Objekte als Rückgabewert ablegt. an den Controller zurückgeben. Die aufgerufene URL kann dabei zum Beispiel ein JSON-Objekt sein. Ein Beispiel zur Nutzung des `$http`-Service ist in Listing 4.5 aufgezeigt.

```
$http.get("http://www.example.com/json/daten").then(function(res) {
            $scope.ergebnis = res.data;
});
```

Listing 4.5: Der `$http`-Service greift auf Daten einer URL zu

State Provider führt die unterschiedlichen Views einer Ionic-Anwendung auf (siehe Listing 4.6). Jeder nicht-abstrakten Oberfläche wird dabei eine URL zugewiesen (Zeile 4). Der Pfad zum entsprechenden Template wird angegeben (Zeile 7) und ein zugehöriger Controller definiert (Zeile 8).

```
1 .config(function($stateProvider,$urlRouterProvider) {
2   $stateProvider
3   .state('startseite', {
4     url: '/startseite',
5     views: {
6       'menuContent': {
7         templateUrl: 'templates/startseite.html',
8         controller: 'StartseiteController'
9     } } } ) }
```

Listing 4.6: Der *$stateProvider* führt die Parameter eines Views auf

Weitere Informationen zu Datenstrukturen in AngularJS finden sich im AngularJS Developer Guide [20].

Im nächsten Kapitel wird die mobile Anwendung implementiert. Dabei wird speziell auf die hier aufgeführten Anforderungen eingegangen.

5

Implementierung und Umsetzung

Dieses Kapitel beschreibt die Umsetzung der Eigenentwicklung. Hierbei wird speziell auf die Implementierung der in Kapitel 4.1 aufgezeigten Anforderungen an das System eingegangen. Dies umfasst sowohl die Aktionen, die von einem Dozenten innerhalb der Webplattform ausgeführt werden, als auch die Aktionen, welche ein Student innerhalb der mobilen Anwendung ausführt.

Vorerst müssen die in der Vorarbeit verwendeten Begriffe `Repository` und `Organization` (siehe [1, Kapitel 4.1.1]) erläutert und voneinander abgegrenzt werden. Ein Repository entspricht einer Veranstaltung, also beispielsweise einer Vorlesung, Übung oder einem Tutorium, welche von einem Dozenten oder wissenschaftlichen Mitarbeiter durchgeführt wird. Lediglich in einem *Repository* können Slides und darin anschließend Fragen erstellt werden. Im weiteren Verlauf wird für ein *Repository* ausschließlich der Begriff *Vorlesung* verwendet.

Eine *Organization* hingegen ist einem Repository übergeordnet. Beispiel für eine *Organisation* ist die Universität Ulm. Diese hat weitere Unterorganisationen, zum Beispiel Fakultäten wie `Ingenieurwissenschaften, Informatik und Psychologie` oder `Mathematik und Wirtschaftswissenschaften`. Diesen sind wiederum Institute wie `Datenbanken und Informationssysteme` untergeordnet. Organisationen sollen dazu dienen, die verschiedenen Vorlesungen sinnvoll und strukturell anzuordnen, damit Dozenten Vorlesungen entsprechend einordnen und diese später von Studierenden rasch gefunden werden können.

In Abbildung 5.1 ist zur besseren Übersicht der Ablauf mit den für die Implementierung wichtigen Schlüsselpunkten aufgeführt. Oberhalb der jeweiligen Funktionen ist das

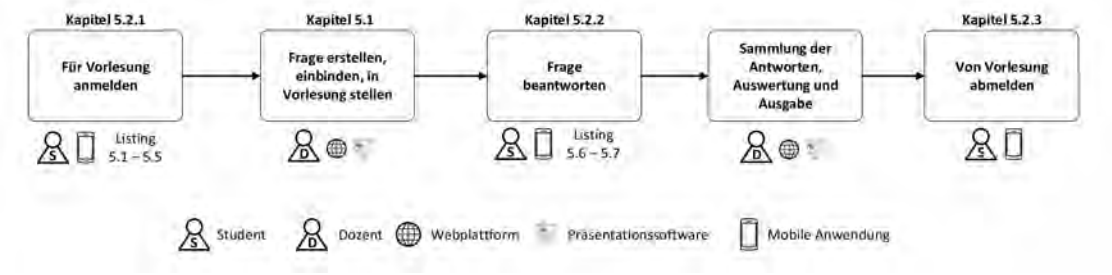


Abbildung 5.1: Ablauf und Struktur der implementierten Funktionen

entsprechende Unterkapitel aufgeführt, bei welchem auf die Umsetzung der jeweiligen Funktion eingegangen wird. Unterhalb wird auf die Codebeispiele (Listings) verwiesen.

5.1 Weiterentwicklung der Webplattform

Dieser Abschnitt beschreibt die Weiterentwicklung der Webplattform und somit den Teil der Arbeit, mit deren Hilfe ein Dozent eine Frage erstellt oder bearbeitet.

Die Webplattform muss um bestimmte Komponenten, wie beispielsweise die Unterstützung weiterer Fragetypen (siehe Kapitel 4.2), erweitert werden. Hierauf wurde bereits in der Vorarbeit (siehe [1, Kapitel 6.1.4]) Stellung bezogen und entsprechend implementiert, dass für das Anlegen eines neuen Fragetyps jeweils drei Dateien angelegt werden müssen. Hierbei handelt es sich um ein `QuestionModel`, welches ein neues Question Object aus den angegebenen Daten erstellt. Des Weiteren muss eine `QuestionView`, welche die Oberfläche und somit die Eingabemaske des Fragetyps darstellt, angelegt werden. Diese enthält zum Beispiel je nach Fragetyp Elemente zur Antwortauswahl wie Radio-Buttons (für eine True-False-Question) oder auch Freitextfelder (für eine Multiple-Answer-Question). Zuletzt benötigt jeder Fragetyp noch eine `Language File`, welches für die unterschiedlichen Komponenten und Attribute eines Fragetyps sprachabhängige Bezeichner bereitstellen.

5.2 Enwicklung der mobilen Anwendung

Dieser Abschnitt beschreibt die Umsetzung und Implementierung der mobilen Anwendung, also jenen Teil der Arbeit, mit welchem die Studierenden innerhalb der mobilen Anwendung mit dem ARS interagieren können. Dabei kann ein Student mit Hilfe der mobilen Anwendung verschiedene Aufgaben ausführen, welche bereits in Kapitel 4.1 aufgeführt wurden.

5.2.1 Anmeldung für eine Vorlesung

Die Struktur der Organisationen ist so aufgebaut, dass eine Anfrage auf die URL `http://SERVER-URL/json/organizations` die oberste Struktur der Organisationen, also sozusagen die Basis-Organisationen, ausgibt. Wenn diese URL erweitert wird, zum Beispiel `http://SERVER-URL/json/organizations/1`, so werden alle Unter-Organisationen der Organisation mit der `ID=1` angezeigt. Die Organisationen werden somit in einer Baumstruktur angelegt (wie bereits zu Beginn von Kapitel **??** erläutert).

Jede Organisation kann jedoch auch Vorlesungen enthalten. Diese werden über die Adresse `http:/SERVER-URL/json/organizations/ID/repositories` aufgerufen, also über den Aufruf der entsprechenden Organization mit dem Anhang `/repositories`.

```
1  .controller('OrganizationsCtrl', function($scope, OrganizationService, organizationData, $ionicLoading) {
2    $scope.organizationsArray = [];
3    $ionicLoading.show();
4    OrganizationService.getOrganizations().then(function(res) {
5      $scope.organizationsArray = res;
6      $ionicLoading.hide();
7    });
8    $scope.organization = {};
9    $scope.setOrganizationActive = function(organization) {
10     organizationData.updateData(organization);
11   }
12   $ionicLoading.hide();
13 })
```

Listing 5.1: Der Controller für die Organizations

Der Controller für die Organizations (`OrganizationCtrl`) in Codebeispiel 5.1 legt ein Array an (Zeile 2) und füllt dieses mit dem Ergebnis des in Codebeispiel 5.2 aufgeführten `OrganizationService`.

Codebeispiel 5.2 zeigt, wie ein Zugriff über den `$http`-Service und eine REST-Anfrage mit Hilfe des Befehls `GET` (siehe Zeile 7) die dort bereitgestellten Informationen, welche im JSON-Format vorliegen, zurückliefert. Sollte es beim Zugriff auf diese URL zu einem Fehler kommen, so wird eine ensprechende Fehlermeldung ausgegeben (siehe Zeile 10-11).

```
1  .factory('OrganizationService', ['$http','$q',function($http,$q){
2      var Url = "server-url";
3      var organizationsUrl = "http://" + Url + "/json/organizations/";
4      return {
5          getOrganizations:function() {
6              var organizations = $q.defer();
7              $http.get(organizationsUrl).then(function(res) {
8                  console.dir(res.data.organizations);
9                  organizations.resolve(res.data.organizations);
10             }, function(error) {
11               alert("There is a problem with your connection. Please check your internet connection!");
12               console.log(error);
13             });
14             return organizations.promise;
15         } }; }])
```

Listing 5.2: Der OrganizationService liest die Organizations aus

Die Darstellung der Organisationen innerhalb der mobilen Anwendung ist in Codebeispiel 5.3 dargestellt. Dabei wird die AngularJS-Methode `ng-repeat` verwendet. Diese bekommt eine Liste in Form eines Arrays übergeben. Anschließend erstellt es aus jedem Objekt dieser Liste ein entsprechendes Objekt innerhalb der View. In diesem Fall `organization in organizationArray` (siehe Zeile 1). Jede Organisation wird dabei durch ihren Namen (`{{organization.name}}`, Zeile 3) und die Beschreibung (`{{organization.description}}`, Zeile 5) repräsentiert. Ein Klick auf die jeweilige Organization ruft deren untergeordnete Organizations und Vorlesungen auf (`href='#/app/organizations/{{organization.id}}'`, Zeile 1).

```
1 <ion-item ng-repeat="organization in organizationsArray" href='#/app/organizations/{{organization.id}}' ng-click="
      setOrganizationActive(organization)">
2       <div class="item-text-wrap">
3         {{organization.name}}
4       </div>
5       <p style="color:#BBBBBB;font-weight:300;" class=" ">{{organization.description}}</p>
```

Listing 5.3: Erzeugung einer Liste der Organizations

Ein Beispiel für die Darstellung der Organisationen mit Hilfe der AngularJS-Methode `ng-repeat` ist in Abbildung 5.2 aufgezeigt.

Die Vorlesungen, wie auch die Organisationen, werden mit Hilfe der Methode `ng-repeat` innerhalb einer Liste dargestellt. Die Implementierung erfolgt analog zur Darstellung der Organisationen, weshalb hier auf ein erneutes Codebeispiel verzichtet wird.

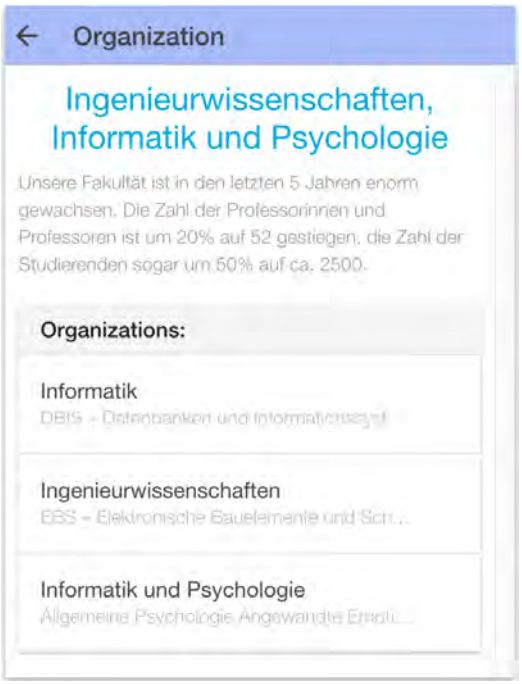


Abbildung 5.2: Darstellung der Organizations als Liste durch AngularJS-Methode `ng-repeat`

Für das Senden von Push-Benachrichtigungen muss das mobile Endgerät eines Studenten eindeutig identifiziert werden. Hierfür wird der Universally Unique Identifier (kurz: UUID) eines mobilen Endgerätes ausgelesen. Im Codebeispiel 5.4 wird die UUID und

das Modell eines mobilen Endgerätes in der Funktion `getDeviceUUID` ausgelesen und übergeben. Es wird dabei zwischen den Plattformen Android und iOS unterschieden (siehe Zeile 4 und Zeile 7). Zu Testzwecken wird in Zeile 10 eine Ausnahmebedingung abgehandelt, falls die mobile Anwendung zum Beispiel auf einem Computer getestet wird. Ein Computer mit Windows oder OS X liefert beispielsweise keine UUID, weshalb hier das Modell `windows` und die UUID `windowstest` verwendet werden. Somit ist ein Test auf einer anderen Plattformen möglich, ohne dass die Anwendung bei der Abfrage der UUID abbricht.

```
1  .service('deviceService', function($cordovaDevice) {
2    return {
3      getDeviceUUID: function() {
4        if(ionic.Platform.isAndroid()){
5          model = $cordovaDevice.getModel();
6          uuid = $cordovaDevice.getUUID();
7        }else if(ionic.Platform.isIOS()){
8          model = $cordovaDevice.getModel();
9          uuid = $cordovaDevice.getUUID();
10       }else{
11         console.log("You're not testing on a provided mobile device. The UUID will be windowstest.");
12         model = "windows";
13         uuid = "windowstest";}
14       return uuid;
15     } } })
```

Listing 5.4: Auslesen der UUID eines mobilen Endgerätes

Um einen Studierenden einer Vorlesung zuzuordnen, wird die UUID seines mobilen Endgerätes zusammen mit der Identifikationsnummer der Vorlesung an das System übertragen (siehe Codebeispiel 5.5). Wieder wird eine REST-Anfrage an den Server gestellt, diesmal jedoch mit einem `POST`-Befehl, welcher eine URL aufruft, dieser jedoch, im Gegensatz zu einer `GET`-Anfrage, auch entsprechende Attribute übergibt (siehe Zeile 6). Dadurch wird eine Zuordnung vom mobilen Endgerät des Studierenden zu einer Vorlesung hergestellt wird.

```
1  $scope.signup = function(device_uuid,activeRepository) {
2    $ionicLoading.show();
3    $scope.uuid = device_uuid;
4    repID = repositoryData.getData();
5    reID = repID.id;
6    $http.post("http://server-url/json/repositories/signup", {"device-id": uuid, "repository-id": reID});
7    $ionicLoading.hide(); };
```

Listing 5.5: Anmeldung zu einer Vorlesung

5.2.2 Beantwortung einer Frage

Kommt der Dozent innerhalb seiner Präsentation auf jene Folie, welche er zuvor mit einer Frage verknüpft hat, wird eine Push-Benachrichtigung an alle Studierenden versendet, welche sich zuvor für diese Vorlesung innerhalb der mobilen Anwendung angemeldet haben.

Die Question-ID wird dabei an das mobile Endgerät des Studierenden übertragen. Mit Hilfe dieser kann über eine REST-Anfrage, gleich wie beim Auslesen der Organisationen oder Vorlesungen, ein JSON-Objekt zurückgeliefert werden. Wie ein solches JSON-Objekt aussehen kann, ist in Codebeispiel 5.6 aufgezeigt. Dabei werden der Typ (Zeile 1), der Fragetext (Zeile 2), die Antwortmöglichkeiten (Zeile 3-5) sowie die richtige(n) Antworte(n) (Zeile 6-7) übergeben.

```
1 { "type": "MultipleAnswerQuestion",
2   "question":"Wofuer steht SQL nicht?",
3   "posibilities": [ "Short Query Language",
4                     "Structured Query Language",
5                     "Semantic Query Language" ],
6   "answer": [ "Short Query Language",
7               "Semantic Query Language" ] }
```

Listing 5.6: Beispiel für ein JSON-Objekt einer Frage

Je nach Fragetyp wird eine entsprechende Vorlage (Template) innerhalb der mobilen Anwendung geladen. Anschließend wird diese Vorlage mit den entsprechenden Informationen, also Fragetext und Antwortmöglichkeiten, aus dem JSON-Objekt gefüllt. Die Antworten werden, analog zur Darstellung der Organisationen und Vorlesungen auch, mit Hilfe der AngularJS-Methode `ng-repeat` dargestellt.

Beim Absenden der Antwort durch den Studierenden wird ein JSON-Objekt an den Server gesendet. Dieses beinhaltet die `device-id`, die `question-id` und den `json_payload`, welcher sowohl den `type` (Fragetyp) als auch die `answer` (Antwort) des Studierenden enthält. Die Funktion, welche die Antwort eines Studierenden absendet, ist in Codebeispiel 5.7 dargestellt.

```
1 $scope.submit = function(device_uuid,activeQuestion) {
2     $ionicLoading.show();
3     uuid = device_uuid;
4     qID = activeQuestion.id);
5     json_payload = {"type": + question.type + ,"answer": + question.answer};
6     $http.post("http://server-url/json/answer", {"device_id": uuid, "question_id": qID, "json_payload": json-payload
          });
7     $ionicLoading.hide(); };
```

Listing 5.7: Submit-Funktion, welche die Antwort eines Studierenden an den Server überträgt

Das Audience Response System überträgt zu keiner Zeit personenbezogene Daten. Eine Ausnahme ist die UUID des Gerätes. Diese identifiziert aber lediglich ein mobiles Endgerät und nicht den Studierenden selbst. Die UUID wird jedoch zu keiner Zeit innerhalb der Webplattform für einen Dozenten sichtbar gemacht. Beim Auslesen der Antworten, also zum Beispiel bei der Darstellung innerhalb des PowerPoint-Plugins, wird ebenfalls eine REST-Anfrage gestellt. Hierbei wird jedoch lediglich der `json_payload` der einzelnen Antworten übergeben. Dieser beinhaltet nur den Typ einer Frage, sowie die Antwort, welche ein Studierender abgegeben hat. Also kann ohne größere Veränderungen am Code der Webplattform über eine REST-Anfrage nicht die UUID ausgelesen und somit ein Rückschluss auf einen einzelnen Studierenden stattfinden. Die Anonymität des Studierenden und seiner Antworten wird gewahrt.

5.2.3 Abmeldung von einer Vorlesung

Beim Abmelden von einer Vorlesung verhält es sich ähnlich wie beim Anmelden. Auch hier wird die UUID eines Gerätes ausgelesen und zusammen mit der Vorlesungs-ID an den Server geschickt. Dieser führt dann eine Löschanfrage in der Datenbank durch und löst die Verknüpfung der UUID (also dem mobilen Endgerät) zur entsprechenden Vorlesung auf. Der Studierende ist somit nicht mehr für diese Vorlesung angemeldet und erhält bei der Aktivierung einer Frage innerhalb dieser Vorlesung keine Push-Benachrichtigung mehr auf sein mobiles Endgerät.

Im nächsten Kapitel wird ein Fazit über die Ausarbeitung gezogen. Dabei wird darauf eingegangen, welche Konzepte betrachtet wurden, was schließlich davon umgesetzt

wurde und wo es zu Problemen bei der Umsetzung kam. Des Weiteren findet ein Ausblick statt, welche Funktionen für das umgesetzte Audience Response System die Zukunft noch denkbar wären.

6

Zusammenfassung

Dieses Kapitel gibt einen Überblick über die Funktionen, welche in dieser Arbeit umgesetzt wurden. Außerdem werden weitere Funktionen und Konzepte angesprochen, mit denen das System noch erweitert werden könnte.

In der vorliegenden Arbeit wurde ein bereits vorhandenes Audience Response System um eine mobile Anwendung erweitert. Dabei wurde zuerst ein Anwendungsfall beschrieben (Kapitel 2) und verwandte Projekte untersucht, ob sie die geforderten Anforderungen und Aufgaben dieses Anwendungsfalles durchführen können (Kapitel 3). Anschließend wurde das verwendete Framework Ionic im Bezug auf Funktionsumfang und Tools genauer betrachtet und ein Anforderungskonzept erarbeitet (Kapitel 4). Mit dieser Grundlage wurde anschließend die mobile Anwendung für die Plattformen Android und iOS umgesetzt (Kapitel 5). Dabei wurde auch die bereits vorhandene Webplattform um weitere Fragetypen erweitert.

Ionic ist noch ein recht junges Framework. Dies lies sich bei der Implementierung unter anderem durch den stetig ansteigenden Umfang an Funktionen, als auch eine überschaubare Anzahl an Tutorials erkennen. Da Ionic jedoch auf Cordova und Angular (Version 1) aufbaut, lassen sich auch Methoden und Funktionen aus diesen Webframeworks nutzen und leicht einbinden.

Generell steht der Nutzung eines hybriden Frameworks zur Implementierung einer mobilen Anwendung nichts im Wege. Es kam zu keiner Zeit zu spürbaren Performanceeinbußen im Vergleich zu ähnlichen nativen Anwendungen. Der Vorteil, bereits bekannte Webtechnologien zu verwenden, lässt einen schnellen Einstieg in ein Framework zu und gute Ergebnisse können früh erzielt werden.

Bei der Nutzung eines Audience Response Systems in der Universität steht am Ende immer auch eine entsprechende Reaktion des Dozenten aus. Sollte durch die abgegebenen Antworten erkennbar sein, dass Studierende Inhalte nicht verstanden haben, kann der Dozent entsprechend reagieren. Er sollte eine Wiederholung des Themas anbieten oder zumindest den Studierenden einen Hinweis geben, wie und wo sie diese entsprechende Themen nachholen können oder wo sie Literatur dazu finden.

6.1 Ausblick

Problematisch ist, dass nicht alle Studierende vom Angebot profitieren können. Zwar wird mit den beiden mobilen Plattformen Android und iOS ein großer Nutzerkreis abgedeckt (etwa 99,1% der Smartphone-Nutzer weltweit [7]), jedoch sind Studierende mit weniger genutzten mobilen Betriebssystemen wie zum Beispiel Windows Phone vom Beantworten der Fragen ausgeschlossen. Außerdem ist für Studierende ohne Smartphone das Beantworten der Fragen nicht möglich.

Zum Zeitpunkt der Ausarbeitung befindet sich Version 2 von Ionic im Beta-Entwicklungsstadium (Stand November 2016). Ionic 2 soll unter anderem eine Unterstützung für Windows Phone bieten. Bereits in der aktuellen Beta-Version lässt sich diese Funktionalität nutzen. Andere Frameworks, zum Beispiel das in Kapitel 2.3 angesprochene Xamarin, bieten bereits eine Unterstützung für die Plattformen Android, iOS und Windows Phone. Zum aktuellen Zeitpunkt gibt es jedoch viele verschiedene hybride Frameworks, welche sich unter anderem im Funktionsumfang unterscheiden. Ein Vergleich dieser und die Wahl eines anderen Frameworks könnte die mobile Anwendung innerhalb des Audience Response Systems durch die Unterstützung weiterer Funktionen noch verbessern.

Bei der Betrachtung der verwandten Projekte in Kapitel 3 fiel auf, dass diese fast immer eine Webplattform zur Beantwortung der Fragen nutzen. Eine solche Plattform bietet den Vorteil, dass ohne die vorherige Installation einer mobilen Anwendung die Teilnahme mit jedem mobilen Endgerät mit Browserunterstützung möglich wird. Damit wäre auch die Unterstützung von (mobilen) Betriebssystemen mit geringeren Nutzerzahlen möglich, ohne dass für jede dieser Plattformen eine eigene Anwendung erstellt werden

muss. Die im vorherigen Punkt angesprochene Umsetzung für Windows Phone würde somit entfallen. Durch die Umsetzung als Webplattform wäre jedoch auch das Problem gegeben, dass ein Student, falls er mehr als ein mobiles Endgerät besitzt, mehrfach an der Beantwortung einer Frage teilnehmen und somit das Ergebnis verfälschen könnte. Außerdem würde hier eine Identifikation über beispielsweise die IP-Adresse oder Cookies statt über eine UUID notwendig werden.

Das umgesetzte Audience Response System unterstützt als Präsentationssoftware lediglich Microsoft PowerPoint. Ein entsprechendes Add-In für Apple Pages oder Google Slides würde Dozenten eine höhere Flexibilität bei der Wahl der Präsentationssoftware für Veranstaltungen bieten. Hierbei ist speziell Google Slides interessant, da es sich um eine kostenlose Präsentationssoftware handelt. Diese lässt sich auch im Browser ausführen und ist somit unabhängig vom genutzten Betriebssystem. Zusammen mit einem Konzept zur Beantwortung der Fragen im Browser wäre das System vollständig im Browser nutzbar. Vorbild ist hier das Audience Response System *Poll Everywhere* (vorgestellt in Kapitel 3). Dieses bietet ein Chrome-Plugin für Google Slides. Nutzer können sich in der Präsentationssoftware in ihren *Poll Everywhere* Account einloggen. Dabei können die Fragen direkt in innerhalb der Präsentationssoftware erstellt, bereits vorhandene Fragen eingefügt und die Auswertung in der Präsentation angezeigt werden.

Literaturverzeichnis

[1] Schwab, F.: Design and Implementation of an Audience Response System for Smart Mobile Devices. Master's thesis, Ulm University (2015)

[2] Schobel, J., Schickler, M., Pryss, R., Nienhaus, H., Reichert, M.: Using vital sensors in mobile healthcare business applications: Challenges, examples, lessons learned. In: International Conference on Web Information Systems and Technologies. (2013) 509–518

[3] : Adobe PhoneGap. (http://phonegap.com/) zuletzt aufgerufen am 15.09.2016.

[4] iDangero.us: Framework7- full featured mobile html framework for building ios & android apps. (https://framework7.io/) zuletzt aufgerufen am 15.09.2016.

[5] Drifty: Ionic: Advanced HTML5 Hybrid Mobile App Framework. (http://ionicframework.com/) zuletzt aufgerufen am 15.09.2016.

[6] : Xamarin - Mobile App Development & App Creation Software. (https://www.xamarin.com/) zuletzt aufgerufen am 15.09.2016.

[7] Gartner, S.D.S.P.I.: Marktanteile der führenden Betriebssysteme am Absatz von Smartphones weltweit vom 1. Quartal 2009 bis zum 2. Quartal 2016. (http://de.statista.com/statistik/daten/studie/73662/umfrage/marktanteil-der-smartphone-betriebssysteme-nach-quartalen/) zuletzt aufgerufen am 15.09.2016.

[8] : ARSnova. (https://arsnova.thm.de/blog/) zuletzt aufgerufen am 04.10.2016.

[9] : Poll Everywhere. (https://www.polleverywhere.com/) zuletzt aufgerufen am 15.09.2016.

[10] : Poll Everywhere - Plans & Pricing. (https://www.polleverywhere.com/plans/higher-ed) zuletzt aufgerufen am 15.09.2016.

[11] : Socrative. (http://www.socrative.com/) zuletzt aufgerufen am 15.09.2016.

[12] : Socrative Pricing. (http://www.socrative.com/pricing) zuletzt aufgerufen am 15.09.2016.

[13] : Mentimeter. (https://www.mentimeter.com/) zuletzt aufgerufen am 15.09.2016.

[14] : Mentimeter Pricing. (https://www.mentimeter.com/plans) zuletzt aufgerufen am 15.09.2016.

[15] Lynch, M.: Announcing The Ionic Framework (The Official Ionic Blog). (http://blog.ionic.io/announcing-ionic/) zuletzt aufgerufen am 15.09.2016.

[16] : Ionic Market - Starters. (https://market.ionic.io/starters) zuletzt aufgerufen am 19.09.2016.

[17] : Ionic Documentation Overview. (http://ionicframework.com/docs/overview/) zuletzt aufgerufen am 19.09.2016.

[18] : AngularJS API Docs. (https://docs.angularjs.org/api) zuletzt aufgerufen am 19.09.2016.

[19] : AngularJS Controllers. (http://www.w3schools.com/angular/angular_controllers.asp) zuletzt aufgerufen am 19.09.2016.

[20] : AngularJS Developer Guide. (https://docs.angularjs.org/guide) zuletzt aufgerufen am 19.09.2016.

Abbildungsverzeichnis

Name: Moritz Weiger Matrikelnummer: 748858

Erklärung

Ich erkläre, dass ich die Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Ulm, den .....

Moritz Weiger